REMARKS

Status of Claims

Claims 1, 3, 12, 14, 19, 21, 23, 26, 31, 34-36 have been amended.

Claim Rejections -35 USC § 102(b)

The Office Action States:

> 4b. **Claims 1-6, 10-15, 19, 20, and 34-35,** are rejected under 35 U.S.C. 102(e) as being anticipated by Grun (US pat. 6,629,166).

However, the Office Action has failed to make a prima facie case of anticipation for the claims, and such, the applicant respectfully submits that the rejections should be withdrawn. "[F]or anticipation under 35 U.S.C. 102, the reference must teach *every aspect* of the claimed invention …" MPEP 706.02 (emphasis added). "The identical invention must be shown *in as complete detail as contained in the ... claim*." *Richardson v., Suzuki Motor Co*., 868 F. 2d 1226, 1236, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989) (emphasis added).

Applicant's amended claim 1 includes, "receiving from a processor a **plurality of write transactions**, which are **identified** by processor **as write combinable** write transactions; storing data …to form **write combined data in response to the plurality of transaction being identified as write combinable write transactions,**" (redacted claim 1 with emphasis). The Office Action points to col. 12 lines 10-12 of Grun, which discloses a format for a message primitive, i.e. a buffer ID field and a service connection field. In addition, the Office Action states the creation of this primitive in an initiator is analogous to write combined data of applicant's claim 1. However, as can be seen, in claim 1, **a plurality of write transactions** are received from a processor. Note that a write transaction, in one embodiment that is described in paragraph 0021 of applicant's disclosure, include a packet, such as a data structure having a header (address, ID, etc.) and a data

payload.

Grun only describes the format for a single message and physical movement of data for "the" message (singular usage of message) are "left to a physical implementation of a target service interface (col. 12 lines 15-20). In stark contrast, applicant's claim 1 includes a plurality of write transactions. The data from the plurality of transactions are stored in a buffer to form "write combined data." As can be seen, write combined data does not include formulating a traditional packet (message, request, command, or data) as suggested in The Office Action, but rather the combining of data from a plurality of write transactions, which may include a plurality of packets.

In addition, Grun does not disclose identifying primitives or messages as write combinable, as described in applicant's claim 1, i.e. write transactions "are identified by the processor as write combinable." Note in paragraph 0020 of applicant's specification, an embodiment is described where "processor 16 does not have to actually combine the writes…the writes are sent out as write combinable." As a result, a processor may transmit the plurality of write transactions identifying them as write combinable and allow an I/O hub to perform the write combining. As discussed above, Grun only discloses sending a single message, selecting a pointer for control of a buffer, and once a complete message is received, passing the pointer back to an I/O device (col. 11 lines 36-39). No where does Grun discuss identifying transactions as write combinable or storing a plurality of messages to form combined data, as described in applicant's claim 1.

Applicant's claim 12 includes, "to receive a first write transaction and a second write transaction from a processor, the first and the second write transactions t**o reference partial data**…storage logic…**to store the partial data**…**as write combined data**," (claim 12 redacted with emphasis). As stated above, Grun only discloses assembling a single message and transmitting the single message utilizing pointers to buffers including the message. However, Grun does not disclose messages or transaction that reference partial data. Furthermore, Grun does not disclose

the partial data from separate messages/transactions being stored as write combined data.

Applicant's claim 34 has been amended to include, "page table logic to identify a page in a memory; wherein the page table logic is to associate a write combining attribute to indicate that partial writes from the page are combinable." Grun does not disclose or suggest use of attributes to identify whether writes from a page of memory are combinable, as disclosed in applicant's claim 34.

Claim Rejections -35 USC § 103(a)

"The examiner bears the initial burden of factually supporting any prima facie conclusion of obviousness." MPEP § 2142. It is well established that *prima facie* obviousness is only established when three basic criteria are met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. *In re Vaeck*, 947 F.2d 488 (Fed. Cir. 1991) (MPEP 2144).

The Office Actions states:

15. **Claims 7-9, 16-18, 31-33, and 36**, are rejected under 35 U.S.C. 103(a) as being unpatentable over Grun (US pat. 6,629,166) in view of Graham et al. (US pat. 6,223,641).

Applicant's claim 31 includes, "determining whether a latency condition exists, the latency condition **including a delay in receiving a next combinable write transaction** from the processor and an interface to the I/O device being in an idle state," (claim 31 emphasis added). As stated in The Office Action, Grun does not disclose determining whether a latency condition exists. Additionally, Graham only discloses the following:

> In a delayed operation, the device generating the command (known as the "master") will receive an acknowledgment that the command has been executed, i.e., that the addressed memory locations have been written with the data delivered with the command, or that the addressed memory locations have been read. Thus, a delayed operation can be used, for example, when the master must wait until the requested operation is completed, before proceeding to the next operation. As just one example, a delayed operation could be used when the master writes a memory location and immediately reads back that memory location for processing.
>
> The master initiates an operation by delivering a command to the bus. In response, the device receiving the command (known as the "target") receives and queues the command for execution, and determines the command will or will not be a delayed operation. If the target determines the command will be a delayed operation, the target returns a Retry message over the PCI bus. Having received the Retry message, the master will subsequently and regularly resubmit the delayed operation command to the PCI bus. So long as the queued command has not be executed by the target, the target will continue to return Retry messages to the master. However, once the target has executed the command, the target will respond appropriately to the resubmitted delayed operation command from the master.

As can be seen, Graham's disclosure of a delayed operation merely includes an operation that delays execution until an acknowledgment is received (i.e. master waiting until the operation is complete). Graham uses a write example, where "once the target has executed the command, the

target will response appropriately." Note that Graham does not disclose whether the operation is a combinable write, as in claim 31. Furthermore, Graham's disclosed target queues the command and responds only when the command is performed, i.e. data is written. In contrast, the latency condition of claim 31 is in regards to a "next combinable write." For example, assume two combinable writes have been received. Here, a latency condition, in one described embodiment, includes an amount of time, execution cycles, or other delay associated with not receiving a third (next) combinable write. In this example, the difference between Graham and claim 31 may be seen, as the delay in Graham is based on waiting for a current write to be performed (data to be written to memory), while the latency condition in claim 31 includes a delay in receiving a next combinable write (not performance of any current writes, but rather the reception of another write to be combined).

The Office Action further states:

22. **Claims 21- 30**, are rejected under 35 U.S.C. 103(a) as being unpatentable over Grun (US pat. 6,629,166) in view of Kelly et al. (US pub. 2004/0019729).

Applicant's claim 21 includes, "a processor **to identify a plurality of write transactions as write combinable**... a write combining module... to store t**he partial data... to form a write combined data set** in response to the plurality of write transactions being identified as write combinable." As stated above, Grun does not disclose identifying write transactions as write combinable or storing partial data to form write combined data, as in applicant's claim 21. In addition, Kelly, which also does not disclose the elements discussed above, was cited for disclosure of a PCI-X bus. As a result, the combination of Grun and Kelly do not teach all the elements of applicant's claim 21.

Therefore, Applicants respectfully submit that claims 1, 12, 21, 31, and 34 are in condition

for allowance, and furthermore, that dependent claims 2-11, 13-20, 22-30, 32-33, and 35-36 are also in condition for allowance for at least the same reasons stated above.

If there are any additional charges, please charge Deposit Account No. 50-0221. If a another telephone interview would in any way expedite the prosecution of the present application , the Examiner is invited to contact David P. McAbee at (503) 712-4988.

Respectfully submitted,
Intel Corporation

Dated: February 13, 2008

/David P. McAbee/Reg. No. 58,104
David P. McAbee
Reg. No. 58,104

Intel Corporation
M/S JF3-147
2111 NE 25th Avenue
Hillsboro, OR 97124
Tele – 503-712-4988
Fax – 503-264-1729